

# **LIS Self Teaching Package 2026**

**R version**

## **Part I**

### **Inequality, poverty, and social policy**





# Part I: Inequality, poverty, and social policy

## **Overall Plan and Structure of the Exercise**

The next eight exercises demonstrate the use of the LIS data. These exercises will lead you through the process of developing a comparative research project that examines inequality and poverty across countries. Each of the exercises introduces new concepts related to the datasets and the programming techniques needed to make use of the data. By the end of the last exercise, you will produce a complete program that returns results on poverty and inequality for a selection of five LIS countries.

Each exercise builds on the one that comes before it. It is intended that you will begin every new activity by returning to the program you have written in the previous exercise, and modifying it to satisfy the requirements of the new exercise. Each exercise contains questions that you can answer with the new results you produce. The solutions<sup>1</sup> available for each exercise always include an example program.

The analysis shown here is simplified somewhat, compared to what might be done in an actual LIS Working Paper. Some choices have also been made in order to demonstrate particular aspects of the LIS data. However, these exercises provide a starting point for researchers who want to develop an analysis of the data based on their own research questions.

## **Research Questions**

Since the beginning of the LIS project, one of the most prominent objects of research using the data has been the effect of government tax and transfer programs on poverty and inequality. The first substantive paper in the LIS Working Papers series, published in 1985, analysed the pre- and post-transfer poverty rate in Sweden, the United Kingdom, Israel, the United States, Norway, Canada, and West Germany.<sup>2</sup> The second substantive paper compares the

---

<sup>1</sup> Please note that results calculated by you might differ from the ones presented here given that the LIS data is subject to updates from time to time.

<sup>2</sup> Richard Hauser, Lee Rainwater, Martin Rein, Gaston Schaber, Timothy Smeeding. "Poverty in Major Industrialized Countries". LIS Working Paper No. 2 – Jul 1985. Available at: <http://www.lisdatacenter.org/wps/liswps/2.pdf>

distribution of income across these same seven countries.<sup>3</sup>

States affect the income distribution through several different types of policy, which are captured in the LIS data:

- Income taxes and social insurance contributions
- Social insurance programmes linked to employment, such as public pensions, unemployment insurance benefits, and sickness pay
- Universal benefits provided irrespective of employment, income or assets
- Social assistance benefits for especially needy individuals or households

The exercises in this section assess the impact of these policies in different countries on both poverty and income inequality. We will measure income from labour, capital, private pensions, and private transfers. Then, we will compare them to income after income taxes and social insurance contributions as well as government transfers are accounted for. Within the category of government policies, we will separate the effect of payroll taxes, social insurance, and universal benefits from the effect of social assistance benefits. We will measure the proportion of the population that is poor according to these different income measures, in which poverty is defined relative to median level of income within a country. We will also compare income inequality using one of the most popular and longstanding inequality measures, the Gini coefficient.

As LIS has grown, the analysis of government policy, poverty, and inequality has been updated for more countries and more recent years.<sup>4</sup> Recently, LIS expanded beyond the rich countries that have long made up the core of the project, and began adding data from middle income countries. This gives researchers the opportunity to compare income and poverty in these countries to the patterns seen in rich countries that have been much more heavily studied.

For this exercise, we will begin by analysing the data from Guatemala. We will then compare Guatemala to four other countries: The United States, Denmark, Hungary, and Israel. These countries have been chosen in part for pedagogical reasons. As well as, they allow for substantively interesting comparisons as they

---

3 Michael O'Higgins, Gunther Schmaus, Geoffrey Stephenson. "Income Distribution and Redistribution". LIS Working Paper No. 3 – Jun 1985. Available at: <http://www.lisdatacenter.org/wps/liswps/3.pdf>

4 For a recent example, see Timothy Smeeding, "Government Programs and Social Outcomes: The United States in Comparative Perspective". LIS Working Paper No. 426 – May 2005. Available at: <http://www.lisdatacenter.org/wps/liswps/426.pdf>

represent a wide range of national income levels and welfare state regime types.

After completing the final exercise, you will be able to answer the following questions:

- Which of the five countries in the study have the highest levels of income inequality and poverty before taxes and transfers are accounted for? Does this change when taxes and transfers are included?
- Which type of government policy has a larger impact on inequality and poverty in each country: taxes, social insurance, and universal benefits, or targeted social assistance?

### **Before you begin**

Before you begin the exercises, take a look at the [2019 Template LIS User Guide](#), which can be accessed through *LIS Website* → *Our Data* → *LIS Database*. The User Guide provides an overview of the structure of LIS data and some data management practices, such as missing values policy and aggregation rules, which will be useful in working with LISSY.

In addition to this, an overview of the datasets and variables is provided through the METadata Information System ([METIS](#)) without having to login to LISSY. You can access METIS via the *LIS Website* → *METIS* → *Enter METIS* → *LIS*. After selecting the datasets and variables, consult the *Results tab* for information on variables and definitions, dataset-specific information and variable availability across datasets.

## 0. The R Package ``lissyrtools``

Following a revision in early 2026, the coding of the exercises in this guide adopts the structure and syntax of the `lissyrtools` package, developed by the LIS team in late 2025. This ensures full consistency with the tools currently used for producing LIS-based aggregated indicators and examples.

`lissyrtools` is an R package designed to simplify and standardize the use of LIS and LWS microdata. It provides a coherent workflow for loading harmonized datasets, transforming variables, applying LIS-consistent weights, and computing key socioeconomic indicators across countries and over time.

The package emphasizes clarity and reproducibility: outputs are compact, country-wise, and easily convertible into tidy, plot-ready data. By handling common methodological steps internally, `lissyrtools` allows users to focus on analysis and interpretation rather than data wrangling and implementation details.

## Contents

### **1. Accessing the LIS databases LISSY Interface**

- Submit a LIS job and get basic descriptive statistics for one variable

### **2. Sample selection and weighting**

- Introduce the list of variables to be used
- Select the sample and eliminate cases with missing data
- Produce weighted and unweighted descriptive statistics for variables

### **3. Working with household income variables: top and bottom coding and equivalence scales**

- Top and bottom coding income data to remove outliers
- Correcting income for household size using equivalence scales

### **4. Inequality: The Gini Index**

- Calculating the Gini coefficient

### **5. Relative poverty rates**

- Calculating relative poverty

### **6. Comparing income concepts**

- Introducing four concepts of income
- Inequality and poverty before and after taxes and transfers

### **7. Comparing multiple countries**

- Extending the analysis to multiple countries
- Working with net vs. gross income data

### **8. Producing compact and concise output**

- Produce easy-to-use output

### **9. Producing graphs with LISSY**

- Generate Lorenz curves for multiple countries

# 1. Accessing the LIS Database: LISSY Interface

## Goal

This exercise introduces [LISSY Interface](#), which we will be using to work with LIS data in all of the subsequent exercises.

LISSY secure web-based interface allows researchers to:

- write, submit and view job requests (and corresponding outputs);
- track the status of the job requests in process ('received', 'processing', 'set for review', 'refused', etc.); and
- access the history of all job requests ever sent.

In this exercise we will use [LISSY interface](#) to open a dataset and produce basic descriptive statistics.

## Activity

Go to *LIS website* → [Login LISSY](#) tab with your LISSY account. Submit a simple program to display descriptive statistics (number of valid observations, mean, minimum, maximum) for the household-level income variable **dhi**, for Guatemala 2006. **dhi**, or disposable household income, contains the total monetary and non-monetary current income for the household, net of income taxes and social security contributions. It is a harmonised variable that is available for all datasets.

## *Questions*

1.1. How many valid observations (non-missing) are in this dataset for **dhi**?

## Guidelines

➤ Once connected to LISSY Interface, there are three main tasks that may be carried out:

1. Submit jobs through the Job Session window >> edit job pane.

- Select a project (LIS, LWS).
- Select a statistical package (SAS, SPSS, Stata or **R**).
- When submitting a job (edit job pane), always add a subject line.
- Write your code.
- Click on the submit button.

2. Work with Today's Jobs (Recent Jobs window.)

- Watch the status of jobs currently sent to LISSY in the 'recent jobs'

pane.

- View the jobs returned by LISSY.
- Click on the received job (marked with green arrow).
- Click on the 'job text' or 'listing' tabs, respectively, of the right panel to see the request and its output.
- Download the received results and the job code by either:
  - clicking on 'save all results to pdf file' button in the middle of the upper tool bar to save the results in pdf format.
  - Clicking on 'save all results to a zipped file' button to save the results in txt format or png format for the produced graphs.
- Re-submit a selected job by clicking on the 'upload all or selected text to the actual job editor' button at the left hand side of the upper tool bar.

### 3. Manage (view, clean and search) all job requests ever sent in the Job Archive window.

- View jobs sent over a specific time period.
- Clean the library by discarding useless job requests (remove job from active archive button).
- Search jobs by keywords.
- Re-submit a selected job by clicking on the 'upload all or selected text to the actual job editor' button at the left hand side of the upper tool bar.

- When you open a LIS dataset, use the correct file reference for the country/year you wish to use. For example:

**lis\_list ← lissytools::lissyuse(data = "gt06")**

For more information about the syntax of country/year file reference, see the job submission instructions on the LIS web site (*Data Access → Job Submission*), accessed through this [link](#).

For a list of available data sets and their 2digit country codes, go to:

*Our Data → LIS Database → List of Datasets*, accessible through [LIS](#)

*Our Data → LWS Database → List of Datasets*, accessible through [LWS](#)

- R reminders

- To access a variable within a data frame, you can use either the syntax **lis\_list [[1]]\$<variable>** or **lis\_list [[1]][['<variable>']]**
- You can get basic descriptive statistics for a variable (such as the

minimum, maximum, mean and median) using `summary(<variable>)`. The option `digits` is needed to ensure that the display precision is great enough, `summary(<variable>, digits=<#>)`.

- To get the total number of non-missing observations of a variable, you can use `sum(!is.na(<variable>))`. This creates a vector which is TRUE whenever the variable is not missing, and then gives you the sum of that vector, i.e., the number of non-missing observations.
- If you do not receive your job in the expected amount of time, it means that there is a long queue of jobs on LISSY. In that case, resending your job, or sending several other ones while waiting to receive the first one, will only increase the queue and hence your waiting time. Remember to wait to get your results before sending a new job!

## **Program**

```
library(lissyrtools)
library(tidyverse) # set of packages that includes amongst others: 'dplyr' and 'purrr'.
```

```
list_gt = lissyrtools::lissyuse(
  # lissyuse imports the selected data frames into a single list
  data = "gt06",
  vars = "dhi"
)
names(list_gt)
df = list_gt[["gt06h"]] # or list_gt[[1]]

print(summary(df$dhi))
nrow(df)
print(sum(!is.na(df$dhi))) # Valid observations of dhi
```

## **Results**

|     | Number of valid observations | Mean   | Minimum  | Maximum   |
|-----|------------------------------|--------|----------|-----------|
| dhi | 13,664                       | 40,670 | -186,010 | 2,729,298 |

### *Solution*

1.1. How many valid observations (non-missing) are in this dataset for **dhi**?

- There are 13,664 valid observations.

### *Comments*

- It is important to pay close attention to sample sizes in order to make sure you have enough data to make stable estimates. When working with small datasets, or small sub-samples of datasets, always check the size of the sample underlying each statistic you have computed.
- As you can see in the results, the disposable household income **dhi** can be negative. This happens in cases where the data provider kept losses as such rather than applying bottom coding techniques. This typically happens with incomes from self-employment or capital income; in rare circumstances, it happens that taxes are higher than gross income due to different income reference periods or miscalculation of taxes.

## 2. Sample selection and weighting

### Goal

This exercise introduces the variables that we will be using in the rest of our analysis and concentrates on sample selection and the use of weights.

*Sample selection* – The final objective of this exercise is to compare incomes before and after government intervention. In order to be sure that the comparison is correct, users should ensure that they use exactly the same sample when calculating statistics for the pre- and post-government intervention. It is thus important to begin by selecting a sample that can be used for the entire analysis. For this reason, we will drop from the analysis not only cases which have missing values in the variable of interest to the specific statistic being calculated at each step, but all the cases that have a missing value in any of the variables of interest for the whole analysis.

*Use of weights* - Comparative researchers are typically interested in the characteristics of national populations, not the samples provided. It is very important to understand and use sample weights correctly in order to get representative results for the total underlying population. This exercise shows the differences in statistics between the unweighted sample and the weighted population.

### Activity

As in the previous exercise, we will continue working with the Guatemala 2006 (GT06) data. Modify your previous program to select only the following variables in addition to **dhi**: household weight (**hpopwgt**), number of household members (**nhhmem**), gross or net income information (**grossnet**), factor income (**hifactor**), work-related insurance transfers (**hpub\_i**), universal benefits (**hpub\_u**), social assistance benefits (**hpub\_a**), private transfers (**hiprivate**) and tax and social security contribution expenditures (**hxitsc**).

(1) Produce two different sets of descriptive statistics for the variables you have selected:

- For continuous variables, the statistics - unweighted, and weighted by person - should include the number of observations, the mean, the median, the minimum and the maximum;
- For categorical variables, you should produce a frequency table.

(2) Drop all cases for which **dhi** or any of its income and expenditure sub-components is missing and see how many cases have missing data.

## Questions

- 2.1. What currency unit is used for the income variables in this dataset?
- 2.2. What effect does applying the weights have on the median of disposable household income?
- 2.3. What percentage of cases is being dropped from the dataset?
- 2.4. In subsequent exercises, we will compare incomes before and after accounting for different kinds of government transfers. What is the difference, in terms of definition, between work-related insurance transfers, universal benefits, and assistance benefits?
- 2.5. In subsequent exercises, we will separate social assistance transfers from work-related insurance and from universal transfers. Based on the information you have so far, which type of transfer do you think has a larger effect on inequality and poverty measures in Guatemala?
- 2.6. How many different values does the variable **grossnet** take in this dataset? How might this variable be useful?

## **Guidelines**

- The income concepts used for LIS income variables are defined in the Interactive METadata Information System (METIS).
- Note to R users: R is a very flexible language, and there are typically many ways to code the same operation. The tips in these lessons suggest one method of achieving some of the exercises goals, but if you are familiar with other techniques you should feel free to use them.
  - *Note that for these early lessons, which concentrate on simple descriptive statistics, R may be somewhat more cumbersome to use than other statistical packages which are designed to make such simple estimates very easy to retrieve. This is in contrast to the later lessons, where R's power and flexibility make it possible to produce more complex estimates very quickly and compactly.*

## **Program**

```
library(lissyrtools)
library(tidyverse)

variables_ex2 <- c(
  "dhi",
  "hifactor",
  "hpub_i",
  "hpub_u",
  "hpub_a",
  "hiprivate",
  "hxitsc"
)

lis_data <- lissyuse("gt06", vars = c(variables_ex2, "grossnet"))
names(lis_data)

# Categorical variables

lissyrtools::run_weighted_count(
  data_list = lis_data,
  var_name = "currency",
  wgt_name = NULL
)
lissyrtools::run_weighted_count(
  data_list = lis_data,
  var_name = "grossnet",
  wgt_name = NULL
)

# Continuous variables

# Unweighted, all cases
```

```

for (x in variables_ex2) {
  cat("Variable:", x, "\n")
  print(summary(lis_data[["gt06h"]][[x]]))
  cat("\n")
}

```

# Note: Although useful for exploratory analysis, the median from summary() relies on R's default quantile method

# rather than the LIS percentile methodology used for official figures.

# Weighted, all cases

```

lis_data_new_wgt <- lis_data %>%
  purrr::map(
    ~ .x %>%
      dplyr::mutate(new_wgt = hpopwgt * nhhmem)
  )

```

# Note: When we are using an HOUSEHOLD level file but want to weight by person rather than household,

# we can multiply the household weight by the number of household members, contained in the variable nhhmem.

# Note: In the `lissyrtools` package the user will find fully integrated weight handling for all functions computing aggregated figures.

```

for (x in variables_ex2) {
  cat("Weighted Average Variable:", x, "\n")

  print(
    lissyrtools::run_weighted_mean(
      data_list = lis_data_new_wgt,
      var_name = x,
      wgt_name = "new_wgt"
    )
  )
}

```

```

)

cat("\n")
}

for (x in variables_ex2) {
  cat("Weighted Percentiles Variable:", x, "\n")

  print(
    lissyrtools::run_weighted_percentiles(
      data_list = lis_data_new_wgt,
      var_name = x,
      wgt_name = "new_wgt"
    )
  )

  cat("\n")
}

# Dropping Missing Cases of dhi

lis_data_new_wgt_no_missings = lis_data_new_wgt %>%
  purrr::map(
    ~ .x %>%
      dplyr::filter(!is.na(dhi))
  )

for (x in variables_ex2) {
  cat("Weighted Average Variable:", x, "\n")

  print(
    lissyrtools::run_weighted_mean(
      data_list = lis_data_new_wgt_no_missings,

```

```
    var_name = x,  
    wgt_name = "new_wgt"  
  )  
)  
  
cat("\n")  
}  
  
for (x in variables_ex2) {  
  cat("Weighted Percentiles Variable:", x, "\n")  
  
  print(  
    lissyrtools::run_weighted_percentiles(  
      data_list = lis_data_new_wgt_no_missings,  
      var_name = x,  
      wgt_name = "new_wgt"  
    )  
  )  
  
  cat("\n")  
}
```

## Results

|         | number of observations | percent |
|---------|------------------------|---------|
| Valid   | 13,664                 | 99.84   |
| Missing | 22                     | 0.162   |
| Total   | 13,686                 | 100     |

| grossnet                                    | number of observations | percent | cumulative percent |
|---|------------------------|---------|--------------------|
| [120]gross, taxes and contributions imputed | 13,686                 | 100     | 100                |
| Total                                       | 13,686                 | 100     |                    |

### Unweighted, all cases

|           | number of observations | mean   | median | minimum  | maximum   |
|-----------|------------------------|--------|--------|----------|-----------|
| dhi       | 13,664                 | 40670  | 27,092 | -186,010 | 2,729,298 |
| hifactor  | 13,664                 | 39,924 | 24,778 | 0        | 3,866,644 |
| hpub_i    | 13,664                 | 803.5  | 0      | 0        | 134,400   |
| hpub_u    | 13,686                 | 0      | 0      | 0        | 0         |
| hpub_a    | 13,686                 | 695.5  | 0      | 0        | 62,400    |
| hiprivate | 13,664                 | 1,439  | 0      | 0        | 128,020   |
| hxitsc    | 13,664                 | 2,191  | 0      | 0        | 1,137,346 |

### Weighted, all cases

|           | mean      | median |
|-----------|-----------|--------|
| dhi       | 49,556.28 | 33,655 |
| hifactor  | 49,298.82 | 30819  |
| hpub_i    | 777.41    | 0      |
| hpub_u    | 0         | 0      |
| hpub_a    | 755.96    | 0      |
| hiprivate | 1,675.62  | 0      |
| hxitsc    | 2,952.58  | 0      |

## Weighted, missing income cases dropped

|           | mean      | median |
|-----------|-----------|--------|
| dhi       | 49,556.28 | 33,655 |
| hifactor  | 49,298.82 | 30,819 |
| hpub_i    | 777.41    | 0      |
| hpub_u    | 0         | 0      |
| hpub_a    | 757.02    | 0      |
| hiprivate | 1,675.62  | 0      |
| hxitsc    | 2,952.58  | 0      |

### Solution

2.1. What currency unit is used for the income variables in this dataset?

- The currency unit in this dataset is the Guatemalan Quetzal. This information can be found on the LIS METIS documentation system at *Our Data* → *METIS*, accessed through this [link](#). One way to find currency information in [METIS](#) is by *clicking on enter Metis* → *selecting LIS* → *selecting a country* → *Results* → *Dataset information* → *[LIS] Dataset*.
- Each LIS dataset has a variable named "currency". This variable indicates the currency unit used for the income/expenditure variables. This variable is available in both the household and the person data files.

```
library(lissyrtools)
```

```
library(tidyverse)
```

```
list_gt = lissyuse("gt06", vars = "dhi") # currency loaded by default
```

```
df = list_gt[[1]]
```

```
df %>% group_by(currency) %>% summarise(a = n())
```

| currency units     | number of observations |
|--------------------|------------------------|
| [320]GTQ - Quetzal | 13,664                 |

2.2. What effect does applying the weights have on the median of disposable household income?

- The median of unweighted **dhi** is 27,092 Quetzals, while the median of weighted **dhi** is 33,658 Quetzals, suggesting that low-income households are over-represented in the sample.

2.3. What percentage of cases is being dropped from the dataset?

- There are 22 cases with missing data, or 0.16 percent of the total number

of cases in the dataset. Note that you should always be careful if you see a large amount of missing data, as it could bias your estimates.

2.4. In subsequent exercises, we will be comparing incomes before and after accounting for different kinds of government transfers. What is the difference between work-related insurance transfers, universal benefits, and assistance benefits?

- These concepts are defined in the [Interactive METadata Information System \(METIS\)](#).

*LIS Database* → *Select variables* → *Major economic aggregates* → *hpub\_i/hpub\_u/hpub\_a* → *click on the information icon ⓘ* → a pop-up menu will appear displaying all the information related to these variables (definition and comments):

- **Work-related insurance transfers:** Monetary transfers stemming from systems where the eligibility is based on the existence and/or the length of an employment relationship; in most cases the benefits are financed by contributions paid by employers, workers or both, and their amount is usually dependent on either the previous earnings or the previous contributions (not including occupational and voluntary pensions);
- **Universal transfers:** Monetary transfers stemming from public programmes that provide flat-rate benefits to certain residents or citizens, provided that they are in a certain situation, but without consideration of income, employment or assets; note that in some cases the benefit amount may also depend on the other incomes of the individuals, which at the limit may result on some proportion of the population at the upper end of the income distribution to be excluded from receipt;
- **Assistance transfers:** Monetary and non-monetary transfers stemming from public programmes that provide benefits especially targeted to needy individuals or households (i.e. with a strict income or assets test); the amount of the benefits is either flat rate or based on the difference between the recipient income and a standard amount representing the minimum subsistence needs as guaranteed by the government.

2.5. In subsequent exercises, we will separate social assistance transfers from work-related insurance and from universal transfers. Based on the information you have so far, which type of transfer do you think has a larger effect on inequality and poverty in Guatemala?

- Since universal transfers are not available at the level of detail required

for Guatemala's dataset. Therefore, inequality and poverty will only be impacted by social assistance transfers and work-related insurance.

2.6. How many different values does the variable **grossnet** take in this dataset? How might this variable be useful?

- The variable **grossnet** has the same value for every case ([120] gross, taxes and contributions imputed). By looking at this variable, you can learn what type of gross or net income information is contained in this variable, even without looking at the documentation. This variable will also be useful when working with multiple datasets that may contain either gross or net income. This information can be found in [METIS](#) under the *Results* → *Dataset information* → *Code Books* tab.

### 3. Working with household income variables: top and bottom coding and equivalence scales

#### Goal

In order to compare incomes across countries, we need to make sure that our variables are fully comparable. In this exercise, you will apply top- and bottom-codes to remove extreme values. You will then create an *equivalised* income variable that adjusts for household size.

*Top and bottom coding* - Many inequality measures are sensitive to the values at the bottom and/or top of the income distribution, and some are not defined for non-positive values of income (e.g., any measure that calculates a logarithm). Applying top and bottom-codes (often referred to as 'winsorising') will avoid this problem, as well as ensuring comparability between datasets that may have originally had different top- and bottom-coding. . At LIS, we use the interquartile range (IQR) method to estimate income measures for the LIS Key figures and Data Access Research Tool (DART). The method detects extreme values and applies lower and upper boundaries as bottom and top code, smoothing inequality trends within and between countries by consistently reducing the influence of extreme values in the income distributions for inequality measures. The first step required is to define a new log transformed variable disposable household income, the second, is to calculate the log values for the interquartile range. Finally, we use the exponential of the log values in the original income distribution before the log transformation:  $EXP [\log Q1 - 3*(\log Q3 - \log Q1)]$  for the lower boundary and  $EXP [\log Q3 + 3*(\log Q3 - \log Q1)]$  for the upper boundary.<sup>5</sup>

*Equivalence scales* - In order to get measures of poverty and/or income inequality in a population, it is necessary to compare income across different types of households. It is not logical to compare total household income between households of different sizes and composition.

Suppose you observe three levels of income (A, B, and C), where  $A > B > C$ . You cannot state that a household earning A is better off than one earning B unless you know the two households are similar in composition. For example, a family of four adult members receiving A is not necessarily better off than a couple with two children who receive B, and the family receiving B may not be better off than the childless couple receiving C.

For this reason, total household income needs to be adjusted to make it comparable across different households. This exercise gives one example of "equalising" households using one specific equivalence scale.

---

<sup>5</sup> See LIS Technical Working Paper No. 9, accessible [here](#).

## **Activity**

- (1) Keep the code from your previous exercise that you used to drop cases with missing data.
- (2) Create a top- and bottom-coded version of **dhi** using the interquartile range technique
- (3) Create an equivalised version of top- and bottom-coded disposable household income. We will correct for household size by applying the "LIS equivalence scale" (i.e., the square root of the number of household members).
- (4) Create also a measure of per-capita income by dividing household income (un-equivalised, but top- and bottom-coded) by the number of household members.
- (5) Produce summary statistics showing the mean, median, minimum, and maximum of the four income variables.

Using the results returned from LISSY, fill out the following table:

|         | Household income (no top or bottom codes) | Household income | Per capita income | Equivalised income |
|---------|---|------------------|-------------------|--------------------|
| Mean    |   |                  |                   |                    |
| Median  |   |                  |                   |                    |
| Minimum |   |                  |                   |                    |
| Maximum |   |                  |                   |                    |

## *Questions*

- 3.1. Which of these four versions of the income variable contain negative values?
- 3.2. Relative to household income and per capita income, how large are the mean and median of equivalised income?
- 3.3. How does applying the top and bottom code affect the mean and median of household income?

## **Guidelines**

- As we continue to build up to our final program, some of the code from the previous exercises will no longer be necessary. (For example, the code that produced the summary statistics in the previous exercise). You can choose to delete this code from your program in order to make it shorter. However, if

you would like to keep a line code but stop it from being executed, simply place a **#** before it.

- This exercise does not use all of the variables that were used in the previous section. But you should continue to keep all of those variables when you open the dataset as they will be needed in future exercises.
- To equalise income, divide the total household income by the value of the equivalence scale for each observation. Be careful when using weights. Make sure that the weight matches your unit of analysis. Weigh by **hpopwgt** for variables which are intrinsically at the household level (e.g., **dhi**) and by **hpopwgt\*nhhmem** (to account for household size) for variables that are conceptually meaningful at the person level (e.g., per capita and equalised income).

## **Program**

```
library(lissyrtools)
```

```
library(tidyverse)
```

```
lis_data <- lissyuse(  
  data = "gt06",  
  vars = "dhi" # weights, technical variables and household size is imported too by default  
) %>%  
  purrr::map(  
    ~ .x %>%  
      dplyr::filter(!is.na(dhi)) %>%  
      dplyr::mutate(new_wgt = hpopwgt * nhhmem)  
  )
```

```
# Reminder: The data is at the household level because the variable requested inside lissyuse  
(dhi) is a household variable.
```

```
# 1st Case: No top and bottom coding
```

```
run_weighted_mean(  
  data_list = lis_data,  
  var_name = "dhi",  
  wgt_name = "hpopwgt"  
)
```

```
run_weighted_percentiles(  
  data_list = lis_data,  
  var_name = "dhi",  
  wgt_name = "hpopwgt",  
  probs = c(0, 0.5, 1) # min, median, max  
)
```

```
# 2nd Case: With top and bottom coding
```

```
tb_code_lis_data = lis_data %>%  
  lissyrtools::apply_iqr_top_bottom_coding(  
    var_name = "dhi",  
    wgt_name = "hpopwgt",  
    times = 3,
```

```
one_sided = NULL,  
type = "type_4"  
)
```

```
run_weighted_mean(  
  data_list = tb_code_lis_data,  
  var_name = "dhi",  
  wgt_name = "hpopwgt"  
)
```

```
run_weighted_percentiles(  
  data_list = tb_code_lis_data,  
  var_name = "dhi",  
  wgt_name = "hpopwgt",  
  probs = c(0, 0.5, 1) # min, median, max  
)
```

```
# 3rd Case: Per Capita dhi
```

```
tb_code_lis_data_pc <- tb_code_lis_data %>%  
  purrr::map(  
    ~ .x %>%  
      dplyr::mutate(dhi_pc = dhi / nhhmem)  
  )
```

```
run_weighted_mean(  
  data_list = tb_code_lis_data_pc,  
  var_name = "dhi_pc",  
  wgt_name = "new_wgt" # see the guidelines  
)
```

```
run_weighted_percentiles(  
  data_list = tb_code_lis_data_pc,  
  var_name = "dhi_pc",  
  wgt_name = "new_wgt", # see the guidelines  
  probs = c(0, 0.5, 1) # min, median, max  
)
```

```
# 4th Case: Equivalized
```

```
eqv_tb_code_lis_data <- tb_code_lis_data %>%  
  apply_sqrt_equivalisation("dhi")
```

```
run_weighted_mean(  
  data_list = eqv_tb_code_lis_data,  
  var_name = "dhi",  
  wgt_name = "new_wgt" # see the guidelines  
)
```

```
run_weighted_percentiles(  
  data_list = eqv_tb_code_lis_data,  
  var_name = "dhi",  
  wgt_name = "new_wgt", # see the guidelines  
  probs = c(0, 0.5, 1) # min, median, max  
)
```

## **Results**

|         | Household<br>income (no<br>top or bottom<br>codes)<br><b>(dhi)</b> | Household income<br>(top or bottom<br>coded) <b>(dhi_tb)</b> | Per capita<br>income (top or<br>bottom coded)<br><b>(pcdhi_tb)</b> | Equivalised<br>income<br><b>(edhi_tb)</b> |
|---------|--|--|--|---|
| Mean    | 47,681.16  | 47,750.27  | 9,769  | 21,407                                    |
| Median  | 31,096   | 31,096   | 5,766  | 14,007                                    |
| Minimum | -186,010   | 358.21   | 59.7   | 146.24                                    |
| Maximum | 2,729,298  | 2,541,391  | 691,527.1  | 1,270,696                                 |

### *Solution*

- 3.1. Which of these four versions of the income variable contain negative values?
- Only the income variable without top or bottom codes contains negative values, which are removed by applying a bottom-code. This will be important in the next exercise on inequality. The measure of inequality we will use, the Gini coefficient, does not allow negative values. Removing

negative values also allows for the commonly used logarithmic transformation of income.

3.2. Relative to household income and per capita income, how large are the mean and median of equivalised income?

- The mean and median values for equivalised income fall between those for household income and those for per capita income. The equivalising formula of  $\mathbf{dhi}/(\mathbf{nhhmem}^{0.5})$  is a compromise between assigning all individuals their household income ( $\mathbf{dhi}/\mathbf{nhhmem}^0$ ) and assigning them a per capita income ( $\mathbf{dhi}/\mathbf{nhhmem}^1$ ).

3.3. How does applying the top and bottom code affect the mean and median of household income?

- Applying top and bottom codes makes the mean lower but does not affect the median (31,096). Means can be very sensitive to extreme values, so median values are often preferred as a measure of central tendency.

## 4. Inequality: The Gini Index

### Goal

This exercise introduces the Gini index, which is one of the most commonly used income inequality indicators. We will be using the Gini coefficient as our measure of inequality in subsequent exercises, in order to compare inequality across countries and across different concepts of income.

### Activity

**Calculate the Gini index on total disposable income for Guatemala in 2006, using variables created in the previous exercise.**

- (1) Start with your program from the previous exercise, which will drop observations with missing data,
- (2) Apply top and bottom codes,
- (3) Create variables containing equivalised disposable income and disposable income per capita,
- (4) Calculate the Gini coefficient for the winsorised (or bottom- and top-coded) versions of household income, per capita income, and equivalised income, and fill out the following table:

|                  | Household income | Per capita income | Equivalised income |
|------------------|------------------|-------------------|--------------------|
| Gini coefficient |                  |                   |                    |

### *Questions*

- 4.1. Which measure shows greater inequality: household income, per capita income or equivalised income? What does this suggest about the possible relationship between income and household size?

## **Program**

```
library(lissyrtools)
```

```
library(tidyverse)
```

```
lis_data_tb <- lissyuse(
```

```
  data = "gt06",
```

```
  vars = "dhi"
```

```
) %>%
```

```
  purrr::map(
```

```
    ~ .x %>%
```

```
      dplyr::filter(!is.na(dhi)) %>%
```

```
      dplyr::mutate(new_wgt = hpopwgt * nhhmem)
```

```
  ) %>%
```

```
  apply_iqr_top_bottom_coding(
```

```
    var_name = "dhi",
```

```
    wgt_name = "hpopwgt"
```

```
) %>%
```

```
  purrr::map(
```

```
    ~ .x %>%
```

```
      dplyr::mutate(
```

```
        dhi_pc = dhi / nhhmem,
```

```
        dhi_eqv = dhi / sqrt(nhhmem)
```

```
      )
```

```
  )
```

```
run_weighted_gini(lis_data_tb, var_name = "dhi", wgt_name = "hpopwgt")
```

```
run_weighted_gini(lis_data_tb, var_name = "dhi_pc", wgt_name = "new_wgt")
```

```
run_weighted_gini(lis_data_tb, var_name = "dhi_eqv", wgt_name = "new_wgt")
```

## **Results**

|                  | Household income | Per capita income | Equivalised income |
|------------------|------------------|-------------------|--------------------|
| Gini coefficient | 0.498            | 0.527             | 0.489              |

### *Solution*

4.1. Which measure shows greater inequality: household income, per capita income or equivalised income? What does this suggest about the possible relationship between income and household size?

- The Gini for equivalised income is lower than the Gini for household income, which in turn is lower than the Gini for per-capita income. This reflects the fact that poorer households in Guatemala tend to be larger on average than richer households. Because the LIS equivalence scale assumes some economies of scale in large households, it produces a lower estimate of inequality than a per-capita measure.

## 5. Relative poverty rates

### Goal

In order to calculate any measure of poverty, it is essential to make some assumptions concerning the criteria used to define poverty. The approach used by LIS (and most commonly adopted in the literature), is that of creating a relative poverty line based on the level and distribution of (equivalised) household disposable income in the total population. Households are classified as poor or non-poor on the basis of whether their income is lower or higher than the relative poverty line.

Once poor households are identified, you can create an indicator to help identify the proportion of poor households (or individuals) and to measure the level of poverty. The choice of the indicator used will mainly depend on the purpose of the research. In this exercise, we will calculate the relative poverty rates of households and individuals in the Guatemala 2006 data.

### Activity

- (1) Add code to your program to produce an indicator for poverty.
- (2) Define the poverty line as 50% of the median equivalised income.
- (3) Calculate both the percentage of households in poverty and the head count ratio (defined as the percentage of individuals living in poor households), and complete the following table.

|                       | Households | Individuals |
|-----------------------|------------|-------------|
| Relative poverty rate |            |             |

### *Question*

- 5.1. Are there more poor *households* or more poor *individuals*? What can you infer from this?

### Guidelines

- From this point forward, we will be working exclusively with equivalised income, so the sections of your code relating to per-capita income can now be commented out or removed.
- Whether your poverty rate is the proportion of *individuals* or *households* in poverty depends on which weighting you use. Use **hpopwgt** if you want to measure household poverty, and **hpopwgt\*nhhmem** if you are interested in individual poverty. If you use the individual-level weighting, you will

produce the Head Count Ratio (HCR), which is the percentage of poor individuals in the total population.

## **Program**

```
library(lissyrtools)
library(tidyverse)

lis_data_tb_eqv <- lissyuse(
  data = "gt06",
  vars = "dhi"
) %>%
  purrr::map(
    ~ .x %>%
      dplyr::filter(!is.na(dhi)) %>%
      dplyr::mutate(new_wgt = hpopwgt * nhmem)
  ) %>%
  apply_iqr_top_bottom_coding(
    var_name = "dhi",
    wgt_name = "hpopwgt"
  ) %>%
  apply_sqrt_equivalisation(var_name = "dhi")

# Poverty Line (50% median)

pov_line <- 0.5 *
  run_weighted_percentiles(
    data_list = lis_data_tb_eqv,
    var_name = "dhi",
    wgt_name = "new_wgt",
    probs = 0.5
  )[[1]]

# Household Poverty
```

```
lis_data_tb_eqv %>%  
  purrr::map(  
    ~ .x %>%  
      mutate(poor = if_else(dhi >= pov_line, 0, 100))  
  ) %>%  
  run_weighted_mean("poor", "hpopwgt")
```

# Individual Poverty

```
lis_data_tb_eqv %>%  
  purrr::map(  
    ~ .x %>%  
      mutate(poor = if_else(dhi >= pov_line, 0, 100))  
  ) %>%  
  run_weighted_mean("poor", "new_wgt")
```

## **Results**

|                       | Households | Individuals |
|-----------------------|------------|-------------|
| Relative poverty rate | 22.1%      | 22.5%       |

### *Solution*

- 5.1. Are there more poor households or more poor individuals? What can you infer from this?
- There is a slightly greater proportion of poor individuals than poor households. This is because poor households are larger on average than non-poor households. This is another reason why the use of the equivalence scale is important.

### *Comments*

- The head count ratio (HCR) measures poverty incidence (i.e., the number or proportion of poor people), but gives every person equal weight no matter how far they fall from the poverty line.
- Another measure, the Income Gap Ratio (IGR) measures poverty intensity or depth (how poor are the poor), but one poor person with an income of an amount  $x$  counts the same as two poor people each with an income of  $x/2$ . That is, the IGR measures the average income gap, but not its distribution among the poor.
- There are many other indicators of poverty that may be useful for different purposes. These include, among the most common, the whole family of Foster-Greer-Thorbecke indicators (of which the HCR is only one), the Sen index, the Takayama index, the Clark index, and the Thon index. It is important to note that a country may score better in comparison to a second country when using a particular index, but could score worse if another index was used instead.

## 6. Comparing income concepts

### Goal

Now that we have calculated Gini coefficients and poverty rates based on equivalised household disposable income, we can easily apply this same code to three other income concepts: income before any taxes and government transfers, income after taxes, social insurance, and universal benefit transfers, and income after social assistance transfers. Starting from this exercise, we will also introduce some programming techniques that will make it easier for you to repeat a series of commands several times without having to repeat the code.

*Different income concepts* - The income variable we have been working with, **dhi**, combines multiple income and expenditure flows. It is the sum of labour and capital income, private transfers, private pensions, work-related insurance transfers, universal benefits, and social assistance benefits, minus any taxes and social insurance contributions paid. We will now define three new concepts of income. One of them is income before *any* government redistribution, but including private pensions. The second is income *after* taxes, social insurance, and universal benefits, but *before* social assistance is included. The third one is after social assistance transfers, but before taxes, social insurance, and universal benefits.

By calculating the Gini coefficient and the poverty rate using each of these four income concepts (income before government intervention, income after non-assistance government redistribution, income after social assistance benefits, and income after all government redistribution, i.e. our original disposable household income variable, **dhi**), we gain some insight into the effect of government programmes on inequality and poverty.

*Efficient programming techniques* - This exercise also introduces some programming techniques that allow looping the same code over several variables.

### Activity

As always, begin with the program you developed for the last exercise. Modify it to create three new income variables. The first, **mi**, is the sum of factor income (**hifactor**), private transfers (**hiprivate**) and private pensions (**hi33**). Because we are specifically interested in the role of *government* transfers, we add private transfers and private pensions to our measure of “market income” from labour and capital.

The second, **siti**, adds **mi** together with social insurance transfers (**hpub\_i**) and universal benefits (**hpub\_u**), while subtracting taxes and social contributions paid (**hxitsc**).

The third measure, **sa**, adds together the market income **mi** with social assistance benefits (**hpub\_a**).

The income variable we have been using up to now, disposable household income, adds together the variables contained in **siti** along with social assistance transfers (which are also contained in the variable **hpub\_a**).

Make sure you apply top and bottom codes and the equivalence scale to the new variables.

Write a loop to calculate the Gini coefficient and the poverty rate for all four income variables, based on the code from the previous two exercises. Use it to fill out the table below. Make sure you use the *same* poverty line for all four income variables. That is, the poverty line should be defined as 50 percent of the median equivalised disposable household income, and that same poverty definition should be applied to the other three income variables.

|                  | Before taxes and government transfers (mi) | After taxes, social insurance benefits and universal benefits (siti) | After social assistance benefits (sa) | After taxes and all government transfers (dhi) |
|------------------|--|--|---------------------------------------|--|
| Gini coefficient |  |  |                                       |  |
| Poverty rate     |  |  |                                       |  |

### Question

6.1. Which government instruments have a greater impact on inequality and poverty in Guatemala: taxes/social insurance/universal benefits, or social assistance?

### Guidelines

- Detailed information about income aggregates used in the exercise can be found in [METIS](#): *METIS* → *LIS database* → *select <variables>* → *Results* → *Variable definitions*. In addition to this, it may be useful to consult the current [list of variables](#) on LIS website: *LIS website* → *Our data* → *LIS database* → *list of variables*.
- Note that social assistance, social insurance and universal benefits (**hpub\_a**, **hpub\_i** and **hpub\_u**) together comprise the concept of public transfers. However, the total sum of these benefits may be lower than the total public transfers reported at the higher-level variable **hpublic**. This is because some amounts that do not clearly belong to one of the transfer categories may be reported directly in the variable **hpublic**. This may also be the case for other higher-level income concepts with an exception of the five blocks of the current income that always add up to the total income (labour income, capital

income, pensions, public social benefits and private transfers). For more information, you can consult the aggregation rules in the [LIS User Guide](#).

- You can easily check whether this is the case by summing up the individual components and the higher-level variable.

## **Program**

```
library(lissyrtools)
```

```
library(tidyverse)
```

```
variables_ex6 = c(
```

```
  "dhi",
```

```
  "hi33",
```

```
  "hpub_a",
```

```
  "hpub_i",
```

```
  "hpub_u",
```

```
  "hifactor",
```

```
  "hiprivate",
```

```
  "hxitsc"
```

```
)
```

```
lis_data <- lissyuse(
```

```
  data = "gt06",
```

```
  vars = variables_ex6
```

```
) %>%
```

```
  purrr::map(
```

```
    ~ .x %>%
```

```
      dplyr::filter(!is.na(dhi)) %>%
```

```
      mutate(
```

```
        new_wgt = nhhmem * hpopwgt,
```

```
        mi = hifactor + hiprivate + hi33,
```

```
        siti = mi + hpub_i + hpub_u - hxitsc,
```

```
        sa = mi + hpub_a
```

```
      )
```

```
) %>%
```

```
  apply_iqr_top_bottom_coding(var_name = "dhi", wgt_name = "hpopwgt") %>%
```

```
  apply_iqr_top_bottom_coding(var_name = "mi", wgt_name = "hpopwgt") %>%
```

```
  apply_iqr_top_bottom_coding(var_name = "sa", wgt_name = "hpopwgt") %>%
```

```
  apply_iqr_top_bottom_coding(var_name = "siti", wgt_name = "hpopwgt") %>%
```

```
apply_sqrt_equivalisation(var_name = "dhi") %>%  
apply_sqrt_equivalisation(var_name = "mi") %>%  
apply_sqrt_equivalisation(var_name = "sa") %>%  
apply_sqrt_equivalisation(var_name = "siti")
```

```
inc <- c("mi", "siti", "sa", "dhi")
```

```
# Gini Index
```

```
for (x in inc) {  
  cat("Gini Index - Variable:", x, "\n")
```

```
  print(  
    run_weighted_gini(  
      data_list = lis_data,  
      var_name = x,  
      wgt_name = "new_wgt"  
    )  
  )  
}
```

```
  cat("\n")
```

```
}
```

```
# Relative Poverty Rate
```

```
pov_line <- 0.5 *  
run_weighted_percentiles(  
  data_list = lis_data,  
  var_name = "dhi",  
  wgt_name = "new_wgt",  
  probs = 0.5  
)[[1]]
```

```

for (w in inc) {
  cat("Relative Poverty Rate - Variable:", w, "\n")

  print(
    lis_data %>%
      purrr::map(
        ~ .x %>%
          mutate(poor = if_else(.x[[w]] >= pov_line, 0, 100)) # carefull with the quotes in the
vector inc, and the need to take them out here
      ) %>%
      run_weighted_mean("poor", "new_wgt")
    )

  cat("\n")
}

```

## **Results**

|                  | Before taxes and government transfers (mi) | After taxes, social insurance benefits and universal benefits (siti) | After social assistance benefits (sa) | After taxes and all transfers (dhi) |
|------------------|--|--|---------------------------------------|-------------------------------------|
| Gini coefficient | 0.516                                      | 0.496 *  | 0.509                                 | 0.489                               |
| Poverty rate     | 24.5%                                      | 24.0% *  | 22.9%                                 | 22.5%                               |

\* Calculation excludes universal benefits since it is not filled in gt06.

### *Solution*

- 6.2. Which government instruments have a greater impact on inequality and poverty in Guatemala: taxes/social insurance/universal benefits, or social assistance?
- Taxes, social insurance benefits and universal benefits play a larger role in reducing income inequality in Guatemala in 2006 than social assistance benefits, and the reverse is true for poverty reduction.

## Comments

- When comparing incomes before and after taxes and transfers, be careful when interpreting the meaning of the pre-tax and transfer figure. It is tempting to interpret this number as a representation of the income distribution that would exist in the absence of government programs. However, since outcomes in the private sector are conditioned by the presence or absence of government programs, this is not generally a reasonable inference.
- There are several online databases containing detailed information on social security systems around the world that may be useful in your analysis:
  - The Mutual Information System on Social Protection ([MISSOC](#)), which provides up-to-date information on social protection legislation, benefits and conditions for 28 EU Member States, Iceland, Liechtenstein, Norway and Switzerland (and [MISSCEO](#) for countries outside of the EU's MISSOC network);
  - [Social Security Programs Throughout the World](#) – a biannual publication highlighting the principal features of social security programs in more than 170 countries;
  - Other databases and publications from institutional websites, such as the International Labour Organisation ([ILO](#)), the Organisation for Economic Co-operation and Development ([OECD](#)) and [EUROMOD](#) – the tax-benefit microsimulation model for European Union;
  - National social security websites.

## 7. Comparing multiple countries

### Goal

Now that we have written code to compute all of our statistics of interest, it is time to calculate these quantities for multiple countries. Building up on the previous exercise, we will introduce different programming techniques that break the code into logic sub-routines and generalise our program to loop through multiple datasets.

*Adding more countries* – Before adding a new country/year to an analysis, it is important to check that the dataset in question has all information necessary for the analysis you are performing. In this case, one should carefully check that the income sub-components variables used in the previous exercise are filled for all the new datasets to be introduced (and if not, whether the analysis can be slightly modified to take into account a different situation).

*Efficient programming techniques* – In the previous exercise we have introduced some programming techniques that allowed to loop the same code over several variables. In this exercise, we will introduce some other techniques that allow to organise the code in an efficient way and easily loop over both variables and datasets.

### Activity

- (1) Take the code from the previous exercise and modify it so that it loops through five datasets: Guatemala 2006 (**gt06**), United States 2004 (**us04**), Denmark 2004 (**dk04**), Hungary 2005 (**hu05**) and Israel 2005 (**il05**).
- (2) The code that creates the income variables can be placed in a subroutine that is called from the main loop, as can the code that applies the equivalence scale and the bottom code.

Use your results to fill in the following tables:

### **Gini Coefficient**

| Dataset | Before taxes and government transfers | After taxes, social insurance benefits and universal benefits | After social assistance benefits | After taxes and all transfers |
|---------|---------------------------------------|---|----------------------------------|-------------------------------|
| GT06    |                                       |   |                                  |                               |
| US04    |                                       |   |                                  |                               |
| DK04    |                                       |   |                                  |                               |
| HU05    |                                       |   |                                  |                               |
| IL05    |                                       |   |                                  |                               |

## Poverty Rate

| Dataset | Before taxes and government transfers | After taxes, social insurance benefits and universal benefits | After social assistance benefits | After taxes and all transfers |
|---------|---------------------------------------|---|----------------------------------|-------------------------------|
| GT06    |                                       |   |                                  |                               |
| US04    |                                       |   |                                  |                               |
| DK04    |                                       |   |                                  |                               |
| HU05    |                                       |   |                                  |                               |
| IL05    |                                       |   |                                  |                               |

Keep in mind that even if all cells can technically be constructed, the result may not necessarily be comparable conceptually! Think carefully about whether the dataset you are looking at contains the necessary information to calculate the quantity in each column.

### Question

- 7.1. In what cells does the figure you produced not match the income concept described in the column header?
- 7.2. In which country do government programmes do the most to reduce inequality and poverty, in percentage terms? In which country do they do the least?
- 7.3. In which countries do social assistance benefits do more to reduce poverty than social insurance plus universal benefits and taxes?

### Guidelines

The functions used to compute aggregated figures in `lissyrtools` are designed to handle multiple countries and years simultaneously. By default, these functions return a list split by country. Each element of the list is a numeric vector, with names corresponding to the years available for that country.

To facilitate further use of these results, the `structure_to_plot()` function converts this list into a single, tidy data frame. The restructured output is suitable for export (e.g. as a CSV file) or for direct use as input in tables and charts.

You should use normalized weights, `hwgt` and `pwgt`, which can be useful when performing pooled analysis with multiple countries.

## Program

```
library(lissyrtools)
library(tidyverse)
# Finally we use the potentiality of `lissyrtools`

ccyy_ex7 = c("gt06", "dk04", "us04", "hu05", "il05")
variables_ex7 = c(
  "dhi",
  "hi33",
  "hpub_a",
  "hpub_i",
  "hpub_u",
  "hifactor",
  "hiprivate",
  "hxitsc"
)

lis_data <- lissyuse(
  data = ccyy_ex7,
  vars = variables_ex7
) %>%
  purrr::map(
    ~ .x %>%
      dplyr::filter(!is.na(dhi)) %>%
      mutate(
        new_wgt = nhhmem * hpopwgt,
        mi = hifactor + hiprivate + hi33,
        siti = mi + hpub_i + hpub_u - hxitsc,
        sa = mi + hpub_a
      )
  ) %>%
```

```

apply_iqr_top_bottom_coding(var_name = "dhi", wgt_name = "hpopwgt") %>%
apply_iqr_top_bottom_coding(var_name = "mi", wgt_name = "hpopwgt") %>%
apply_iqr_top_bottom_coding(var_name = "sa", wgt_name = "hpopwgt") %>%
apply_iqr_top_bottom_coding(var_name = "siti", wgt_name = "hpopwgt") %>%
apply_sqrt_equivalisation(var_name = "dhi") %>%
apply_sqrt_equivalisation(var_name = "mi") %>%
apply_sqrt_equivalisation(var_name = "sa") %>%
apply_sqrt_equivalisation(var_name = "siti")

```

```
inc <- c("mi", "siti", "sa", "dhi")
```

```
# Gini Index
```

```
for (x in inc) {
```

```
  cat("Gini Index - Variable:", x, "\n")
```

```
  print(
```

```
    run_weighted_gini(
```

```
      data_list = lis_data,
```

```
      var_name = x,
```

```
      wgt_name = "new_wgt"
```

```
    ) %>%
```

```
    structure_to_plot()
```

```
  )
```

```
  cat("\n")
```

```
}
```

```
# Relative Poverty Rate
```

```
pov_line <- run_weighted_percentiles(
```

```

data_list = lis_data,
var_name = "dhi",
wgt_name = "new_wgt",
probs = 0.5
) # Now we will be dealing with a list

for (w in inc) {
  cat("Relative Poverty Rate - Variable:", w, "\n")

  print(
    purrr::map2(
      # we use map2 to work with two lists, the one with the microdata, and the one with the respective
      # poverty lines
      lis_data,
      pov_line,
      ~.x %>%
        mutate(poor = if_else(.x[[w]] >= .y * 0.5, 0, 100)) # carefull with the quotes in the vector inc, and the
        # need to take them out here
      ) %>%
      run_weighted_mean("poor", "new_wgt") %>%
      structure_to_plot()
    )

  cat("\n")
}

```

## **Results**

### **Gini Coefficient**

| Dataset | Before taxes and government transfers | After taxes, social insurance benefits and universal benefits | After social assistance benefits | After taxes and all transfers |
|---------|---------------------------------------|---|----------------------------------|-------------------------------|
| GT06    | 0.516                                 | 0.496 **  | 0.509                            | 0.489                         |
| US04    | 0.488                                 | 0.399   | 0.468                            | 0.377                         |
| DK04    | 0.422                                 | 0.262   | 0.395                            | 0.230                         |
| HU05    | 0.527 *                               | 0.305   | 0.510                            | 0.288                         |
| IL05    | 0.505                                 | 0.407   | 0.486                            | 0.383                         |

### **Poverty Rate**

| Dataset | Before taxes and government transfers | After taxes, social insurance benefits and universal benefits | After social assistance benefits | After taxes and all transfers |
|---------|---------------------------------------|---|----------------------------------|-------------------------------|
| GT06    | 24.5                                  | 24.0 **   | 22.9                             | 22.5                          |
| US04    | 25.9                                  | 20.7  | 22.4                             | 16.7                          |
| DK04    | 25.3                                  | 12.7  | 22.2                             | 5.6                           |
| HU05    | 42.6 *                                | 9.6   | 41.3                             | 6.7                           |
| IL05    | 28.6                                  | 22.6  | 26.7                             | 19.4                          |

\*Calculation based on post-tax income

\*\*Calculation excludes universal benefits

### *Solution*

7.1. In what cells does the figure you produced not match the income concept described in the column header?

- Because Hungary 2005 is a net income dataset, the Gini before taxes and transfers cannot be included. While the example program below does produce a result for Hungary, it is not actually comparable to the other countries because it is post-tax. That cell should therefore be left blank.

7.2. In which country do government programmes do the most to reduce inequality and poverty, in percentage terms? In which country do they do the least?

- Government programmes have the largest impact on reducing inequality in Denmark, where they reduce the Gini coefficient by 45 percent, and poverty in Hungary by 84 percent.

- Government programs have the lowest impact on reducing inequality and poverty in Guatemala, where they reduce the Gini coefficient by 6 percent and poverty by 8 percent.
- 7.3. In which countries do social assistance benefits do more to reduce poverty than social insurance plus universal benefits and taxes?
- In Guatemala, social assistance reduces poverty more than social insurance plus taxes and universal benefits.

#### *Comments*

- The datasets in these exercises were chosen because they allow social assistance to be separated from other kinds of government transfers. In many datasets, unfortunately, this separation is not possible due to the limitations of the original data. In such cases, the total amount of transfer income will be contained in a higher-level variable such as **hpublic**, and lower level variables such as **hpub\_i** and **hpub\_u** will be unfilled. You can consult the **Crossed-compare** function in the METIS documentation tool, to determine which variables are available in each dataset. This is the **main functionality** of METIS. Upon entering the tool ([METIS](#)), it is possible to view the availability of the selected variable(s) in the selected dataset(s) and to **compare information about this selection**. To access this information, *Select datasets* → *Select variables (under Income Aggregates)* → *Results* → *Crossed-Compare*.

## 8. Producing compact and concise output

### Goal

Compute income inequality and relative poverty rates for multiple income concepts across several country–year pairs using `lissyrtools`, and present the results in a single, tidy table suitable for export and visualization.

### Activity

Using LIS microdata and the `lissyrtools` package:

- Load household-level data for a selected set of countries and years.
- Construct different income concepts (market income, social assistance income, social insurance transfers income, and disposable household income).
- Apply LIS-consistent data treatments, including: top- and bottom-coding using the IQR rule, square-root equalization, appropriate population weights.
- Compute, for each income concept: the Gini index and the relative poverty rate at 50% of the Median.
- Combine all results into a single table and export it as a CSV file.

### Guidelines

Use `lissyuse()` to load all required datasets and variables in one step.

Work with lists of country–year data frames and rely on `purrr` to iterate consistently across them.

Compute the poverty line once, based on equalised disposable household income, and reuse it for all income concepts.

Use `structure_to_plot()` to convert list-based outputs into tidy data frames.

Ensure that all aggregated indicators are computed using the appropriate LIS weights.

The final output should be a tidy table with one row per country, year, and income concept, containing both the Gini index and the relative poverty rate.

## **Program**

```
library(lissyrtools)

library(tidyverse)

# Finally we use the potentiality of `lissyrtools`

ccyy_ex8 = c("gt06", "dk04", "us04", "hu05", "il05")

variables_ex8 = c(
  "dhi",
  "hi33",
  "hpub_a",
  "hpub_i",
  "hpub_u",
  "hifactor",
  "hiprivate",
  "hxitsc"
)

lis_data <- lissyuse(
  data = ccyy_ex8,
  vars = variables_ex8
) %>%

purrr::map(
  ~.x %>%
  dplyr::filter(!is.na(dhi)) %>%
  mutate(
    new_wgt = nhhmem * hpopwgt,
    mi = hifactor + hiprivate + hi33,
    siti = mi + hpub_i + hpub_u - hxitsc,
    sa = mi + hpub_a
  )
) %>%

apply_iqr_top_bottom_coding(var_name = "dhi", wgt_name = "hpopwgt") %>%
```

```

apply_iqr_top_bottom_coding(var_name = "mi", wgt_name = "hpopwgt") %>%
apply_iqr_top_bottom_coding(var_name = "sa", wgt_name = "hpopwgt") %>%
apply_iqr_top_bottom_coding(var_name = "siti", wgt_name = "hpopwgt") %>%
apply_sqrt_equivalisation(var_name = "dhi") %>%
apply_sqrt_equivalisation(var_name = "mi") %>%
apply_sqrt_equivalisation(var_name = "sa") %>%
apply_sqrt_equivalisation(var_name = "siti")

```

```
inc <- c("mi", "siti", "sa", "dhi")
```

```
# Build a Function
```

```
compute_gini_poverty <- function(data, var, pov_line) {
```

```
  gini <- run_weighted_gini(
```

```
    data_list = data,
```

```
    var_name = var,
```

```
    wgt_name = "new_wgt"
```

```
  ) %>%
```

```
  structure_to_plot()
```

```
poverty <- purrr::map2(
```

```
  data,
```

```
  pov_line,
```

```
  ~ .x %>%
```

```
    mutate(poor = if_else(.data[[var]] < .y * 0.5, 100, 0))
```

```
  ) %>%
```

```
  run_weighted_mean("poor", "new_wgt") %>%
```

```
  structure_to_plot()
```

```
dplyr::tibble(
```

```
  variable = var,
```

```
  cname = gini$name,
```

```

year = gini$year,
gini = gini$value,
poverty = poverty$value
)
}

# Compute poverty line once
pov_line <- run_weighted_percentiles(
  data_list = lis_data,
  var_name = "dhi",
  wgt_name = "new_wgt",
  probs = 0.5
)

# Build the final table with purrr::map_dfr and turn into into a csv to print and copy & paste
results_table <- purrr::map_dfr(
  inc,
  ~ compute_gini_poverty(
    data = lis_data,
    var = .x,
    pov_line = pov_line
  )
)

write.csv(results_table)

```

## **Results**

The output at the end of program should resemble the following structure:

```
","variable","cname","year","gini","poverty"  
"1","mi","Denmark",2004,0.422490494389422,25.3496394916183  
"2","mi","Guatemala",2006,0.51639489752688,24.4763834371688  
"3","mi","Hungary",2005,0.526534903503659,42.5628273324202  
"4","mi","Israel",2005,0.505367046244146,28.5842665513769  
"5","mi","United States",2004,0.488235635921228,25.8652486675433  
"6","siti","Denmark",2004,0.262277808928047,12.6993806144169  
"7","siti","Guatemala",2006,0.496235588917898,24.029038952285  
"8","siti","Hungary",2005,0.305254851919813,9.61701337166639  
"9","siti","Israel",2005,0.4074477266131,22.5611062685058  
"10","siti","United States",2004,0.399172781038942,20.6951059599721  
"11","sa","Denmark",2004,0.394673163227708,22.1595496609471  
"12","sa","Guatemala",2006,0.508802003810615,22.8857041841621  
"13","sa","Hungary",2005,0.509882124550359,41.3032650547111  
"14","sa","Israel",2005,0.485433733329273,26.7382203499991  
"15","sa","United States",2004,0.467839633049152,22.4482942081551  
"16","dhi","Denmark",2004,0.22986716311425,5.63544895573962  
"17","dhi","Guatemala",2006,0.488560213909295,22.4896095486091  
"18","dhi","Hungary",2005,0.28772235714024,6.72128767510126  
"19","dhi","Israel",2005,0.382947714095735,19.3604964458996  
"20","dhi","United States",2004,0.376518813653092,16.6527530825389
```

If you copy and paste this into a spreadsheet, most spreadsheet programs should recognize this as a set of comma-separated values and parse it automatically. You can also copy the text into a text file, save it with the **.CSV** extension, and open it with your spreadsheet program.

For example, you can copy and paste the values into an Excel file, use the *Text To Columns* function under the *Data* tab and choose *comma* as a delimiter.

## 9. Producing graphs with LISSY

### Goal

This exercise introduces LISSY's graphing feature that allows users to generate, display and export graphs based on LIS data. Your goal is to visually compare income inequality across different countries, using our variable for disposable household income (**dhi**).

### Activity

In this exercise you will generate a Lorenz Curve on disposable household income (dhi) for Guatemala (gt06), the United States (us04), and Denmark (dk04) in order to compare the degree of income inequality among these countries.

- (1) Prepare the R session, loading the required packages and functions
- (2) Import the files for Guatemala (**gt06**), the United States (**us04**), and Denmark (**dk04**)
- (3) Prepare the data for the analysis, making sure you drop observations with missing data and consider the winsorised (or bottom- and top-coded) version of the equivalised disposable income
- (4) Generate Lorenz Curves Guatemala, United States, and Denmark using the winsorised version equivalised income

### *Question*

- 9.1. Compare the degree of income inequality among Guatemala, the United States and Denmark using Lorenz Curve.

### Guidelines

Once the percentiles have been estimated, we use the `structure_to_plot()` function to convert the output into a clean, tidy data frame.

For the Lorenz curve estimates, we then compute the cumulative sums in order to obtain income shares across the distribution.

In LISSY this step is not mandatory; however, if the user wishes to produce more than one chart, each output should be explicitly assigned a name, as illustrated in the following line:

```
print(png(file = paste0(USR_PDF, "/lorenz",".png"), width = 1280,  
height = 720, res = 150 ))
```

## **Program**

```
library(lissyrtools)
```

```
library(tidyverse)
```

```
library(ggplot2)
```

```
library(ggthemes)
```

```
# Finally we use the potentiality of `lissyrtools`
```

```
ccyy_ex9 = c("gt06", "dk04", "us04")
```

```
lorenz_estimates <- lissyuse(
```

```
  data = ccyy_ex9,
```

```
  vars = "dhi"
```

```
) %>%
```

```
  purrr::map(
```

```
    ~ .x %>%
```

```
      dplyr::filter(!is.na(dhi)) %>%
```

```
      mutate(
```

```
        new_wgt = nhhmem * hpopwgt
```

```
      )
```

```
) %>%
```

```
  apply_iqr_top_bottom_coding(var_name = "dhi", wgt_name = "hpopwgt") %>%
```

```
  apply_sqrt_equivalisation(var_name = "dhi") %>%
```

```
  run_weighted_percentiles(
```

```
    "dhi",
```

```
    "new_wgt",
```

```
    probs = seq(0, 1, 0.01),
```

```
    share = TRUE
```

```
) %>%
```

```
  structure_to_plot() %>%
```

```
  group_by(dname) %>%
```

```
  mutate(
```

```

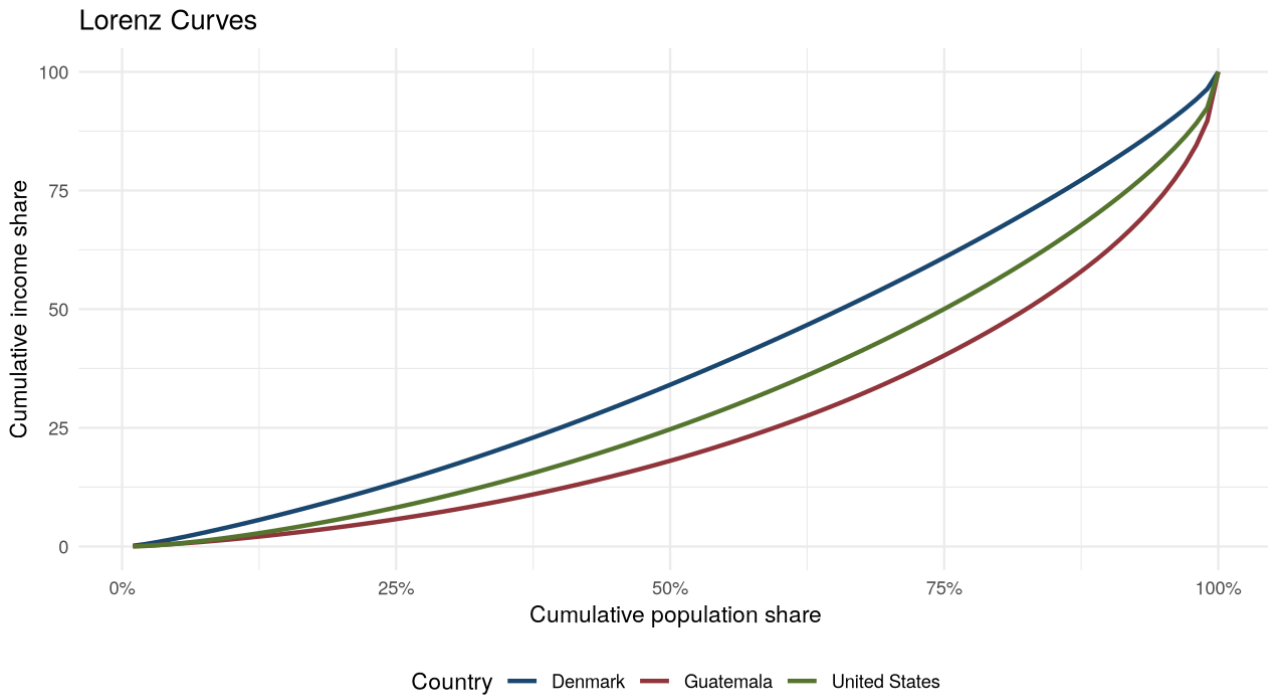
    cumsum = cumsum(value),
    pop_share = row_number() / max(row_number())
) %>%
ungroup()

print(png(file = paste0(USR_PDF, "/lorenz",".png"), width = 1280, height = 720, res = 150 ))
ggplot(lorenz_estimates, aes(x = pop_share, y = cumsum, color = cname)) +
  geom_line(size = 1) +
  scale_x_continuous(labels = scales::percent) +
  labs(
    x = "Cumulative population share",
    y = "Cumulative income share",
    color = "Country",
    title = "Lorenz Curves"
  ) +
  theme_minimal() +
  scale_color_stata() +
  theme(legend.position = "bottom")

```

## Results

**Graph – Lorenz Curve comparison between Guatemala (2006), United States (2004) and Denmark (2004)**



### *Solution*

9.1. Compare the degree of income inequality among Guatemala, United States, and Denmark using Lorenz Curve.

- The Lorenz curve is a classical tool for describing income distributions that allows making pair-wise rankings, in the case in which the curves do not cross.
- Among the countries considered, the United States is the one with highest level of inequality, while Denmark is the one with the lowest.
- It is important to remind that intersecting Lorenz curves prevents conclusions on which income distribution has more inequality.